CS29003 ALGORITHMS LABORATORY
(WorkSheet 1-Solutions)
Date: Sep 12 2020

1 Compute the worst case time complexity

1.1 Question 1

In the outer for-loop, the variable i keeps halving so it goes round $\log_2 n$ times. For each i, next loop goes round also $\log_2 n$ times, because of doubling the variable j. The innermost loop by k goes round n/2 times. Loops are nested, so the bounds may be multiplied to give that the algorithm is $\mathcal{O}(n(\log n)^2)$.

1.2 Question 2 [assume n $=2^m$]

The outer for loop goes round n times. For each i, the next loop goes round m= $\log_2 n$ times, because of doubling the variable j. For each j, the innermost loop by k goes round j times, so that the two inner loops together go round $1 + 2 + 4 + \ldots + 2^{m-1} = 2^m - 1 \approx$ n times. Loops are nested, so the bounds may be multiplied to give that the algorithm is $\mathcal{O}(n^2)$.

1.3 Question 3 [compute the tight bound]

The first and second successive innermost loops have $\mathcal{O}(n)$and $\mathcal{O}(\log n)$ complexity, respectively. Thus, the overall complexity of the inner most part is $\mathcal{O}(n)$. The outermost and middle loops have complexity $\mathcal{O}(\log n)$ and $\mathcal{O}(n)$, so a straightforward (and valid) solution is that the overall complexity is $\mathcal{O}(n^2 \log n)$.

More detailed analysis would show that the outermost and middle loops are interrelated, and the number of repeating the innermost part is as follows:
$1 + 2 + \ldots + 2^m = 2^{m+1} - 1$ where m=$\lfloor \log_2 n \rfloor$ is the smallest integer such that $2^{m+1} > n$. Thus actually this code has quadratic complexity $\mathcal{O}(n^2)$.